9-22-00

A

Please type a plus sign (+) inside this box → +

PTO/SB/05 (4/98)
Approved for use through 09/30/2000 OMB 0651-0032
Patent and Trademark Office U S DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1 53(b))

Attorney Docket No.	042390.P8098X
First Inventor or Application Identifier	Carl M. Ellison
Title	Managing a Secure Platform Using a Hierarchical Executive Architecture in
Express Mail Label No.	EL466331454US

APPLICATION ELEMENTS
See MPEP chapter 600 concerning utility patent application contents

ADDRESS TO: Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

1. ☒ Fee Transmittal Form
(Submit an original, and a duplicate for fee processing)
2. ☒ Specification [Total Pages 37]
(preferred arrangement set forth below)
   - Descriptive title of the Invention
   - Cross References to Related Applications
   - Statement Regarding Fed sponsored R & D
   - Reference to Microfiche Appendix
   - Background of the Invention
   - Brief Summary of the Invention
   - Brief Description of the Drawings *(if filed)*
   - Detailed Description
   - Claim(s)
   - Abstract of the Disclosure
3. ☒ Drawing(s) *(35 U.S.C. 113)* [Total Sheets 13]
4. Oath or Declaration [Total Pages 12]
   a. ☐ Newly executed (original copy)
   b. ☐ Copy from a prior application (37 C.F.R. § 1.63(d)) *(for continuation/divisional with Box 16 completed)*
      i. ☐ DELETION OF INVENTOR(S) Signed statement attached deleting inventor(s) named in the prior application, see 37 CFR §§ 1.63(d)(2) and 1.33(b).

*NOTE FOR ITEMS 1 & 13 IN ORDER TO BE ENTITLED TO PAY SMALL ENTITY FEES, A SMALL ENTITY STATEMENT IS REQUIRED (37 C.F.R. § 1.27), EXCEPT IF ONE FILED IN A PRIOR APPLICATION IS RELIED UPON (37 C.F.R. § 1.28).

5. ☐ Microfiche Computer Program *(Appendix)*
6. Nucleotide and/or Amino Acid Sequence Submission *(if applicable, all necessary)*
   a. ☐ Computer Readable Copy
   b. ☐ Paper Copy (identical to computer copy)
   c. ☐ Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

7. ☒ Assignment Papers (cover sheet & document(s))
8. ☐ 37 C.F.R. § 3.73(b) Statement *(when there is an assignee)* ☐ Power of Attorney
9. ☐ English Translation Document *(if applicable)*
10. ☐ Information Disclosure Statement (IDS)/PTO - 1449 ☐ Copies of IDS Citations
11. ☐ Preliminary Amendment
12. ☒ Return Receipt Postcard (MPEP 503) *(Should be specifically itemized)*
13. ☐ *Small Entity Statement(s) ☐ Statement filed in prior application, Status still proper and desired
14. ☐ Certified Copy of Priority Document(s) *(if foreign priority is claimed)*
15. ☐ Other: ....

16. If a **CONTINUING APPLICATION**, *check appropriate box, and supply the requisite information below and in a preliminary amendment:*
☐ Continuation ☐ Divisional ☒ Continuation-in-part (CIP) of prior application No: 09/539,344
Prior application Information: Examiner ____________ *Group/Art Unit:* ____________

For CONTINUATION or DIVISIONAL APPS only The entire disclosure of the prior application, from which an oath or declaration is supplied under Box 4b, is considered a part of the disclosure of the accompanying continuation or divisional application and is hereby incorporated by reference The incorporation can only be relied upon when a portion has been inadvertently omitted from the submitted application parts.

17. CORRESPONDENCE ADDRESS

☐ *Customer Number of Bar Code Label* *(Insert Customer No. or Attach bar code label here)* or ☒ *Correspondence address below*

Name	BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP				
Address	12400 Wilshire Boulevard, Seventh Floor				
City	Los Angeles	State	California	Zip Code	90025
Country	U.S.A.	Telephone	(714) 557-3800	Fax	(714) 557-3347

Name (Print/Type)	Thinh V. Nguyen, Reg. No. 42,034		
Signature	[signature]	Date	09/22/00

Burden Hour Statement This form is estimated to take 0.2 hours to complete Time will vary depending upon the needs of the individual case Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231 DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS SEND TO Assistant Commissioner for Patents, Box Patent Application, Washington, DC 20231

Please type a plus sign (+) inside this box → +

PTO/SB/17 (12/99)
Approved for use through 09/30/2000 OMB 0651-0032
Patent and Trademark Office U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number

FEE TRANSMITTAL for FY 2000

Patent fees are subject to annual revision
Small Entity payments must be supported by a small entity statement, otherwise large entity fees must be paid. See Forms PTO/SB/09-12. See 37 C.F.R §§ 1.27 and 1.28.

TOTAL AMOUNT OF PAYMENT ($) 1,888.00

Complete if Known	
Application Number	09/539,344
Filing Date	March 31, 2000
First Named Inventor	Carl M. Ellison
Examiner Name	Not Assigned
Group/Art Unit	2785
Attorney Docket No.	042390.P8098

METHOD OF PAYMENT (check one)

1. ☒ The Commissioner is hereby authorized to charge indicated fees to
   ☒ The Commissioner is hereby authorized to credit any over payments to

Deposit Account Number: 02-2666

Deposit Account Name: Blakely, Sokoloff, Taylor & Zafman LLP

☒ Charge Any Additional Fees Required Under 37 CFR §§ 1.16,1.17, 1.18 and 1.20

2. ☒ Payment Enclosed:
   ☒ Check ☐ Money Order ☐ Other

FEE CALCULATION

1. BASIC FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
101	690	201	345	Utility filing fee	$690.00
106	310	206	155	Design filing fee	
107	480	207	240	Plant filing fee	
108	690	208	345	Reissue filing fee	
114	150	214	75	Provisional filing fee	

SUBTOTAL (1) ($) 690.00

2. EXTRA CLAIM FEES

			Extra Claims		Fee from below		Fee Paid
Total Claims	80	- 20 =	60	X	18.00	=	$1,080.00
Independent Claims	4	- 3 =	1	X	78.00	=	$78.00
Multiple Dependent						=	

**or number previously paid, if greater, For Reissues, see below

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description
103	18	203	9	Claims in excess of 20
102	78	202	39	Independent claims in excess of 3
104	260	204	130	Multiple Dependent claim, if not paid
109	78	209	39	**Reissue Independent claims over original patent
110	18	210	9	**Reissue claims in excess of 20 and over original patent

SUBTOTAL (2) ($) 1,158.00

FEE CALCULATION (continued)

3. ADDITIONAL FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
105	130	205	65	Surcharge - late filing fee or oath	
127	50	227	25	Surcharge - late provisional filing fee or cover sheet.	
139	130	139	130	Non-English specification	
147	2,520	147	2,520	For filing a request for reexamination	
112	920*	112	920*	*Requesting publication of SIR prior to Examiner action	
113	1,840*	113	1,840*	*Requesting publication of SIR after Examiner action	
115	110	215	55	Extension for response within first month	
116	380	216	190	Extension for response within second month	
117	870	217	435	Extension for response within third month	
118	1,210	218	680	Extension for response within fourth month	
128	1,850	228	925	Extension for response within fifth month	
119	300	219	150	Notice of Appeal	
120	300	220	150	Filing a brief in support of an appeal	
121	260	221	130	Request for oral hearing	
138	1,510	138	1510	Petition to institute a public use proceeding	
140	110	240	55	Petition to revive - unavoidable	
141	1,210	241	605	Petition to revive - unintentional	
142	1,210	242	605	Utility issue fee (or reissue)	
143	430	243	215	Design issue fee	
144	580	244	290	Plant issue fee	
122	130	122	130	Petitions to the Commissioner	
123	50	123	50	Petitions related to provisional applications	
126	240	126	240	Submission of Information Disclosure Stmt	
581	40	581	40	Recording each patent assignment per property (times number of properties)	40.00
146	790	246	395	Filing a submission after final rejection (37 CFR 1.129(a))	
149	790	249	395	For each additional invention to be examined (37 CFR 1.129(b))	
Other fee (specify)					
Other fee (specify)					

*Reduced by Basic Filing Fee Paid

SUBTOTAL (3) ($) 40.00

SUBMITTED BY

Typed or Printed Name	Thinh V. Nguyen	Date	09/22/00	Complete (if applicable) Reg. Number	42,034
Signature				Deposit Account User ID	02-2666

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Box Patent Application, Washington, DC 20231.

Docket No. 042390.P8098X
Express Mail No.: EL466331454US

UNITED STATES PATENT APPLICATION

FOR

MANAGING A SECURE PLATFORM USING A HIERARCHICAL EXECUTIVE ARCHITECTURE IN ISOLATED EXECUTION MODE

INVENTORS:

CARL M. ELLISON

ROGER A. GOLLIVER

HOWARD C. HERBERT

DERRICK C. LIN

FRANCIS X. MCKEEN

GIL NEIGER

KEN RENERIS

JAMES A. SUTTON

SHREEKANT S. THAKKAR

MILLIND MITTAL

PREPARED BY:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 Wilshire Blvd., 7th Floor
Los Angeles, CA 90025-1026
(714) 557-3800

CROSS-REFERENCES TO RELATED APPLICATIONS

This is a continuation-in-part of U.S. Patent Application No. 09/539,344 filed March 31, 2000.

BACKGROUND

1. Field of the Invention

This invention relates to microprocessors. In particular, the invention relates to processor security.

2. Description of Related Art

Advances in microprocessor and communication technologies have opened up many opportunities for applications that go beyond the traditional ways of doing business. Electronic commerce (E-commerce) and business-to-business (B2B) transactions are now becoming popular, reaching the global markets at a fast rate. Unfortunately, while modern microprocessor systems provide users convenient and efficient methods of doing business, communicating and transacting, they are also vulnerable for unscrupulous attacks. Examples of these attacks include theft of data, virus, intrusion, security breach, and tampering, to name a few. Computer security, therefore, is becoming more and more important to protect the integrity of the computer systems and increase the trust of users.

Threats caused by unscrupulous attacks may be in a number of forms. An invasive remote-launched attack by hackers may disrupt the normal operation of a system connected to thousands or even millions of users. A virus program may corrupt code and/or data of a single-user platform.

Existing techniques to protect against attacks have a number of drawbacks. Anti-virus programs can only scan and detect known viruses. Security co-processors or smart cards using cryptographic or other security techniques have limitations in speed performance, memory capacity, and flexibility. Redesigning operating systems creates software compatibility issues and causes tremendous investment in development efforts.

BRIEF DESCRIPTION OF THE DRAWINGS

The features and advantages of the present invention will become apparent from the following detailed description of the present invention in which:

Figure 1A is a diagram illustrating a logical architecture according to one embodiment of the invention.

Figure 1B is a diagram illustrating accessibility of various elements in the operating system and the processor according to one embodiment of the invention.

Figure 1C is a diagram illustrating a computer system in which one embodiment of the invention can be practiced.

Figure 2 is a diagram illustrating an executive subsystem according to one embodiment of the invention.

Figure 3 is a diagram illustrating a processor executive handler shown in Figure 2 according to one embodiment of the invention.

Figure 4 is a diagram illustrating a processor executive shown in Figure 2 according to one embodiment of the invention.

Figure 5 is a diagram illustrating an operating system executive shown in Figure 2 according to one embodiment of the invention.

Figure 6 is a diagram illustrating a boot-up code shown in Figure 2 according to one embodiment of the invention.

Figure 7 is a flowchart illustrating a process to manage a secure platform according to one embodiment of the invention.

Figure 8 is a flowchart illustrating a process to boot up platform according to one embodiment of the invention.

Figure 9 is a flowchart illustrating a process to execute an isolated create instruction according to one embodiment of the invention.

Figure 10 is a flowchart illustrating a process to handle a processor executive according to one embodiment of the invention.

Figure 11 is a flowchart illustrating a process to handle an operating system executive according to one embodiment of the invention.

DESCRIPTION

The present invention is a method and apparatus to manage a secure platform. A processor executive (PE) handles an operating system executive (OSE) in a secure environment. The secure environment has a platform key (PK) and is associated with an isolated memory area in the platform. The OSE manages a subset of an operating system (OS) running on the platform. The platform has a processor operating in one of a normal execution mode and an isolated execution mode. The isolated memory area is accessible to the processor in the isolated execution mode. A PE supplement supplements the PE with a PE manifest representing the PE and a PE identifier to identify the PE. A PE handler handles the PE using the PK and the PE supplement.

A boot-up code boots up the platform following a power on. The secure environment includes an OSE supplement to supplement the OSE with an OSE manifest representing the OSE and an OSE identifier to identify the OSE. The PE handler includes a PE loader, a PE manifest verifier, a PE verifier, a PE key generator, a PE identifier logger, and a PE entrance/exit handler. The PE loader loads the PE and the PE supplement from a PE memory into the isolated memory area using a parameter block provided by the boot-up code. The PE manifest verifier verifies the PE manifest. The PE verifier verifies the PE using the PE manifest and a constant derived from the PK. The PE key generator generates a PE key using the PK. The PE key generator includes a PE key combiner to combine the PE identifier and the PK. The combined PE identifier and the PK correspond to the PE key. The PE identifier logger logs the PE identifier in a storage. The PE entrance/exit handler handles a PE entry and a PE exit.

The OSE handler includes an OSE loader, an OSE manifest verifier, an OSE verifier, an OSE key generator, an OSE identifier logger, and an OSE entrance/exit handler. The OSE loader loads the OSE and the OSE supplement into the isolated memory area. The OSE manifest verifier verifies the OSE manifest. The OSE verifier verifies the OSE. The OSE key generator generates an OSE key. The OSE identifier logger logs the OSE identifier in a storage. The OSE entrance/exit handler handles an OSE entry and an OSE exit. The OSE key generator includes a binding key generator and an OSE key combiner. The binding key generator generates a binding key (BK) using the PE key. The OSE key combiner combines the OSE identifier and the BK. The combined OSE identifier and the BK correspond to the OSE key.

The OSE includes a module loader and evictor, a key binder and unbinder, a page manager, an interface handler, a scheduler and balancer, and an interrupt handler.

The module loader and evictor loads and evicts a module into and out of the isolated memory area, respectively. The module is one of an application module, an applet module, and a support module. The page manager manages paging in the isolated memory area. The interface handler handles interface with the OS. The key binder and unbinder includes an applet key generator to generate an applet key associating with the applet module. The applet key generator includes an applet key combiner to combine the OSE key with an applet identifier identifying the applet module. The combined OSE key and the applet identifier correspond to the applet key.

The boot up code includes a PE locator, a PE recorder, and an instruction invoker. The PE locator locates the PE and the PE supplement. The PE locator transfers the PE and the PE supplement into the PE memory at a PE address. The PE recorder records the PE address in the parameter block. The instruction invoker executes an isolated create instruction which loads the PE handler into the isolated memory area. The isolated create instruction performs an atomic non-interruptible sequence. The atomic sequence includes a number of operations: a physical memory operation, an atomic read-and-increment operation, an isolated memory area control operation, a processor isolated execution operation, an PE handler loading operation, a PE handler verification, and an exit operation. The physical memory operation verifies if the processor is in a flat physical page mode. The atomic read-and-increment operation reads and increments a thread count register in a chipset. The read-and-increment operation determines if the processor is the first processor in the isolated execution mode. The isolated memory area control operation configures the chipset using a configuration storage. The processor isolated execution operation configures the processor in the isolated execution mode. The processor isolated execution operation includes a chipset read operation and a processor configuration operation. The chipset read operation reads the configuration storage in the chipset when the processor is not a first processor in the isolated execution mode. The processor configuration operation configures the processor according to the configuration storage when the processor is not a first processor in the isolated execution mode. The PE handler loading operation loads the PE handler into the isolated memory area. The PE handler verification verifies the loaded PE handler. The exit operation transfers control to the loaded PE handler.

The chipset includes at least one of a memory controller hub (MCH) and an input/output controller hub (ICH). The storage is in an input/output controller hub (ICH) external to the processor.

In the following description, for purposes of explanation, numerous details are set forth in order to provide a thorough understanding of the present invention. However, it will be apparent to one skilled in the art that these specific details are not required in order to practice the present invention. In other instances, well-known electrical structures and circuits are shown in block diagram form in order not to obscure the present invention.

ARCHITECTURE OVERVIEW

One principle for providing security in a computer system or platform is the concept of an isolated execution architecture. The isolated execution architecture includes logical and physical definitions of hardware and software components that interact directly or indirectly with an operating system of the computer system or platform. An operating system and the processor may have several levels of hierarchy, referred to as rings, corresponding to various operational modes. A ring is a logical division of hardware and software components that are designed to perform dedicated tasks within the operating system. The division is typically based on the degree or level of privilege, namely, the ability to make changes to the platform. For example, a ring-0 is the innermost ring, being at the highest level of the hierarchy. Ring-0 encompasses the most critical, privileged components. In addition, modules in Ring-0 can also access to lesser privileged data, but not vice versa. Ring-3 is the outermost ring, being at the lowest level of the hierarchy. Ring-3 typically encompasses users or applications level and executes the least trusted code. It is noted that the level of the ring hierarchy is independent to the level of the security protection of that ring.

Figure 1A is a diagram illustrating a logical operating architecture 50 according to one embodiment of the invention. The logical operating architecture 50 is an abstraction of the components of an operating system and the processor. The logical operating architecture 50 includes ring-0 10, ring-1 20, ring-2 30, ring-3 40, and a processor nub loader 52. The processor nub loader 52 is an instance of a processor executive (PE) handler. The PE handler is used to handle and/or manage a processor executive (PE) as will be discussed later. The logical operating architecture 50 has two modes of operation: normal execution mode and isolated execution mode. Each ring in

the logical operating architecture 50 can operate in both modes. The processor nub loader 52 operates only in the isolated execution mode.

Ring-0 10 includes two portions: a normal execution Ring-0 11 and an isolated execution Ring-0 15. The normal execution Ring-0 11 includes software modules that are critical for the operating system, usually referred to as kernel. These software modules include primary operating system (e.g., kernel) 12, software drivers 13, and hardware drivers 14. The isolated execution Ring-0 15 includes an operating system (OS) nub 16 and a processor nub 18. The OS nub 16 and the processor nub 18 are instances of an OS executive (OSE) and processor executive (PE), respectively. The OSE and the PE are part of executive entities that operate in a secure environment associated with the isolated area 70 and the isolated execution mode. The processor nub loader 52 is a protected bootstrap loader code held within a chipset in the system and is responsible for loading the processor nub 18 from the processor or chipset into an isolated area as will be explained later.

Similarly, ring-1 20, ring-2 30, and ring-3 40 include normal execution ring-1 21, ring-2 31, ring-3 41, and isolated execution ring-1 25, ring-2 35, and ring-3 45, respectively. In particular, normal execution ring-3 includes N applications 42_1 to 42_N and isolated execution ring-3 includes K applets 46_1 to 46_K.

One concept of the isolated execution architecture is the creation of an isolated region in the system memory, referred to as an isolated area, which is protected by both the processor and chipset in the computer system. Portions of the isolated region may also be in cache memory. Access to this isolated region is permitted only from a front side bus (FSB) of the processor, using special bus (e.g., memory read and write) cycles, referred to as isolated read and write cycles. The special bus cycles are also used for snooping. The isolated read and write cycles are issued by the processor executing in an isolated execution mode when accessing the isolated area. The isolated execution mode is initialized using a privileged instruction in the processor, combined with the processor nub loader 52. The processor nub loader 52 verifies and loads a ring-0 nub software module (e.g., processor nub 18) into the isolated area. The processor nub 18 provides hardware-related services for the isolated execution.

One task of the processor nub loader 52 and processor nub 18 is to verify and load the ring-0 OS nub 16 into the isolated area, and to generate the root of a key hierarchy unique to a combination of the platform, the processor nub 18, and the operating system nub 16. The operating system nub 16 provides links to services in the

primary OS 12 (e.g., the unprotected operating system), provides page management within the isolated area, and has the responsibility for loading ring-3 application modules 45, including applets 46_1 to 46_K, into protected pages allocated in the isolated area. The operating system nub 16 may also load ring-0 supporting modules.

The operating system nub 16 may choose to support paging of data between the isolated area and ordinary (e.g., non-isolated) memory. If so, then the operating system nub 16 is also responsible for encrypting and hashing the isolated area pages before evicting the page to the ordinary memory, and for checking the page contents upon restoration of the page. The isolated mode applets 46_1 to 46_K and their data are tamper-resistant and monitor-resistant from all software attacks from other applets, as well as from non-isolated-space applications (e.g., 42_1 to 42_N), drivers and even the primary operating system 12. The only software that can interfere with or monitor the applet's execution is the processor nub loader 52, processor nub 18 or the operating system nub 16.

Figure 1B is a diagram illustrating accessibility of various elements in the operating system 10 and the processor according to one embodiment of the invention. For illustration purposes, only elements of ring-0 10 and ring-3 40 are shown. The various elements in the logical operating architecture 50 access an accessible physical memory 60 according to their ring hierarchy and the execution mode.

The accessible physical memory 60 includes an isolated area 70 and a non-isolated area 80. The isolated area 70 includes applet pages 72 and nub pages 74. The non-isolated area 80 includes application pages 82 and operating system pages 84. The isolated area 70 is accessible only to elements of the operating system and processor operating in isolated execution mode. The non-isolated area 80 is accessible to all elements of the ring-0 operating system and to the processor.

The normal execution ring-0 11 including the primary OS 12, the software drivers 13, and the hardware drivers 14, can access both the OS pages 84 and the application pages 82. The normal execution ring-3, including applications 42_1 to 42_N, can access only to the application pages 82. Generally applications can only access to their own pages, however, the OS typically provides services for sharing memory in controlled methods. Both the normal execution ring-0 11 and ring-3 41, however, cannot access the isolated area 70.

The isolated execution ring-0 15, including the OS nub 16 and the processor nub 18, can access to both of the isolated area 70, including the applet pages 72 and the

nub pages 74, and the non-isolated area 80, including the application pages 82 and the OS pages 84. The isolated execution ring-3 45, including applets 46_1 to 46_K, can access only applet pages 72. The applets 46_1 to 46_K reside in the isolated area 70. In general, applets can only access their own pages; however, the OS nub 16 can also provides services for the applet to share memory (e.g., share memory with other applets or with non-isolated area applications).

Figure 1C is a diagram illustrating a computer system 100 in which one embodiment of the invention can be practiced. The computer system 100 includes a processor 110, a host bus 120, a memory controller hub (MCH) 130, a system memory 140, an input/output controller hub (ICH) 150, a non-volatile memory, or system flash, 160, a mass storage device 170, input/output devices 175, a token bus 180, a motherboard (MB) token 182, a reader 184, and a token 186. The MCH 130 may be integrated into a chipset that integrates multiple functionalities such as the isolated execution mode, host-to-peripheral bus interface, memory control. Similarly, the ICH 150 may also be integrated into a chipset together or separate from the MCH 130 to perform I/O functions. For clarity, not all the peripheral buses are shown. It is contemplated that the system 100 may also include peripheral buses such as Peripheral Component Interconnect (PCI), accelerated graphics port (AGP), Industry Standard Architecture (ISA) bus, and Universal Serial Bus (USB), etc. The "token bus" may be part of the USB bus, e.g., it may be hosted on the USB bus.

The processor 110 represents a central processing unit of any type of architecture, such as complex instruction set computers (CISC), reduced instruction set computers (RISC), very long instruction word (VLIW), or hybrid architecture. In one embodiment, the processor 110 is compatible with an Intel Architecture (IA) processor, such as the Pentium™ series, the IA-32™ and the IA-64™. The processor 110 includes a normal execution mode 112 and an isolated execution circuit 115. The normal execution mode 112 is the mode in which the processor 110 operates in a non-secure environment, or a normal environment without the security features provided by the isolated execution mode. The isolated execution circuit 115 provides a mechanism to allow the processor 110 to operate in an isolated execution mode. The isolated execution circuit 115 provides hardware and software support for the isolated execution mode. This support includes configuration for isolated execution, definition of an isolated area, definition (e.g., decoding and execution) of isolated instructions, generation of isolated access bus cycles, and access checking.

In one embodiment, the computer system 100 can be a single processor system, such as a desktop computer, which has only one main central processing unit, e.g. processor 110. In other embodiments, the computer system 100 can include multiple processors, e.g. processors 110, 110a, 110b, etc., as shown in Figure 1C. Thus, the computer system 100 can be a multi-processor computer system having any number of processors. For example, the multi-processor computer system 100 can operate as part of a server or workstation environment. The basic description and operation of processor 110 will be discussed in detail below. It will be appreciated by those skilled in the art that the basic description and operation of processor 110 applies to the other processors 110a and 110b, shown in Figure 1C, as well as any number of other processors that may be utilized in the multi-processor computer system 100 according to one embodiment of the present invention.

The processor 110 may also have multiple logical processors. A logical processor, sometimes referred to as a thread, is a functional unit within a physical processor having an architectural state and physical resources allocated according to some partitioning policy. Within the context of the present invention, the terms "thread" and "logical processor" are used to mean the same thing. A multi-threaded processor is a processor having multiple threads or multiple logical processors. A multi-processor system (e.g., the system comprising the processors 110, 110a, and 110b) may have multiple multi-threaded processors.

The host bus 120 provides interface signals to allow the processor 110 or processors 110, 100a, and 110b to communicate with other processors or devices, e.g., the MCH 130. In addition to normal mode, the host bus 120 provides an isolated access bus mode with corresponding interface signals for memory read and write cycles. The isolated access bus mode is asserted on memory accesses initiated while the processor 110 is in the isolated execution mode and it is accessing memory within the isolated area. The isolated access bus mode is also asserted on instruction pre-fetch and cache write-back cycles if the address is within the isolated area address range. The isolated access bus mode is configured within the processor 110. The processor 110 responds to a snoop cycle to a cached address when the isolated access bus mode on the FSB matches the mode of the cached address.

The MCH 130 provides control and configuration of system memory 140. The MCH 130 provides interface circuits to recognize and service isolated access assertions on memory reference bus cycles, including isolated memory read and write cycles. In

addition, the MCH 130 has memory range registers (e.g., base and length registers) to represent the isolated area in the system memory 140. Once configured, the MCH 130 aborts any access to the isolated area that does not have the isolated access bus mode asserted.

The system memory 140 stores system code and data. The system memory 140 is typically implemented with dynamic random access memory (DRAM) or static random access memory (SRAM). The system memory 140 includes the accessible physical memory 60 (shown in Figure 1B). The accessible physical memory includes a loaded operating system 142, the isolated area 70 (shown in Figure 1B), and an isolated control and status space 148. The loaded operating system 142 is the portion of the operating system that is loaded into the system memory 140. The loaded OS 142 is typically loaded from a mass storage device via some boot code in a boot storage such as a boot read only memory (ROM). The isolated area 70, as shown in Figure 1B, is the memory area that is defined by the processor 110 when operating in the isolated execution mode. Access to the isolated area 70 is restricted and is enforced by the processor 110 and/or the MCH 130 or other chipset that integrates the isolated area functionalities. The isolated control and status space 148 is an input/output (I/O)-like, independent address space defined by the processor 110. The isolated control and status space 148 contains mainly the isolated execution control and status registers. The isolated control and status space 148 does not overlap any existing address space and is accessed using the isolated bus cycles. The system memory 140 may also include other programs or data that are not shown.

The ICH 150 represents a known single point in the system having the isolated execution functionality. For clarity, only one ICH 150 is shown. The system 100 may have many ICH's similar to the ICH 150. When there are multiple ICH's, a designated ICH is selected to control the isolated area configuration and status. In one embodiment, this selection is performed by an external strapping pin. As is known by one skilled in the art, other methods of selecting can be used, including using programmable configuring registers. The ICH 150 has a number of functionalities that are designed to support the isolated execution mode in addition to the traditional I/O functions. In particular, the ICH 150 includes an isolated bus cycle interface 152, the processor nub loader 52 (shown in Figure 1A), a digest memory 154, a cryptographic key storage 155, an isolated execution logical processor manager 156, and a token bus interface 159.

The isolated bus cycle interface 152 includes circuitry to interface to the isolated bus cycle signals to recognize and service isolated bus cycles, such as the isolated read and write bus cycles. The processor nub loader 52, as shown in Figure 1A, includes a processor nub loader code and its digest (e.g., cryptographic hash) value. The processor nub loader 52 is invoked by execution of an appropriate isolated instruction (e.g., Iso_Init) and is transferred to the isolated area 70. From the isolated area 80, the processor nub loader 52 copies the processor nub 18 from the system flash memory (e.g., the processor nub code 18 in non-volatile memory 160) into the isolated area 70, verifies and logs its integrity, and manages a symmetric key used to protect the processor nub's secrets. In one embodiment, the processor nub loader 52 is implemented in read only memory (ROM). For security purposes, the processor nub loader 52 is unchanging, tamper-resistant and non-substitutable. The digest memory 154, typically implemented in RAM, stores the digest (e.g., cryptographic hash) values of the loaded processor nub 18, the operating system nub 16, and any other supervisory modules (e.g., ring-0 modules) loaded into the isolated execution space. The cryptographic key storage 155 holds a symmetric encryption/decryption key that is unique for the platform of the system 100. In one embodiment, the cryptographic key storage 155 includes internal fuses that are programmed at manufacturing. Alternatively, the cryptographic key storage 155 may also be created during manufacturing with a cryptographic random number generator. The isolated execution logical processor manager 156 manages the operation of logical processors configuring their isolated execution mode support. In one embodiment, the isolated execution logical processor manager 156 includes a logical processor count register that tracks the number of logical processors participating in the isolated execution mode. The token bus interface 159 interfaces to the token bus 180. A combination of the processor nub loader digest, the processor nub digest, the operating system nub digest, and optionally additional digests, represents the overall isolated execution digest, referred to as isolated digest. The isolated digest is a fingerprint identifying the all supervisory code involved in controlling the isolated execution configuration and operation. The isolated digest is used to attest or prove the state of the current isolated execution environment.

The non-volatile memory 160 stores non-volatile information. Typically, the non-volatile memory 160 is implemented in flash memory. In one embodiment, the non-volatile memory 160 includes the processor nub 18. The processor nub 18 provides set-up and low-level management of the isolated area 70 (in the system

memory 140), including verification, loading, and logging of the operating system nub 16, and the management of the symmetric key used to protect the operating system nub's secrets. The processor nub loader 52 performs some part of the setup and manages/updates the symmetric key before the processor nub 18 and the OS nub 16 are loaded. The processor nub 18 The processor nub 18 may also provide interface abstractions to low-level security services provided by other hardware. The processor nub 18 may also be distributed by the original equipment manufacturer (OEM) or operating system vendor (OSV).

The mass storage device 170 stores archive information such as code (e.g., processor nub 18), programs, files, data, applications (e.g., applications 42_1 to 42_N), applets (e.g., applets 46_1 to 46_K) and operating systems. The mass storage device 170 may include compact disk (CD) ROM 172, floppy diskettes 174, and hard drive 176, and any other storage devices. The mass storage device 170 provides a mechanism to read machine-readable media. When implemented in software, the elements of the present invention are the code segments to perform the necessary tasks. The program or code segments can be stored in a processor readable medium or transmitted by a computer data signal embodied in a carrier wave, or a signal modulated by a carrier, over a transmission medium. The "processor readable medium" may include any medium that can store or transfer information. Examples of the processor readable medium include an electronic circuit, a semiconductor memory device, a ROM, a flash memory, an erasable programmable ROM (EPROM), a floppy diskette, a compact disk CD-ROM, an optical disk, a hard disk, a fiber optical medium, a radio frequency (RF) link, etc. The computer data signal may include any signal that can propagate over a transmission medium such as electronic network channels, optical fibers, air, electromagnetic, RF links, etc. The code segments may be downloaded via computer networks such as the Internet, an Intranet, etc.

I/O devices 175 may include any I/O devices to perform I/O functions. Examples of I/O devices 175 include a controller for input devices (e.g., keyboard, mouse, trackball, pointing device), media card (e.g., audio, video, graphics), a network card, and any other peripheral controllers.

The token bus 180 provides an interface between the ICH 150 and various tokens in the system. A token is a device that performs dedicated input/output functions with security functionalities. A token has characteristics similar to a smart card, including at least one reserved-purpose public/private key pair and the ability to

sign data with the private key. Examples of tokens connected to the token bus 180 include a motherboard token 182, a token reader 184, and other portable tokens 186 (e.g., smart card). The token bus interface 159 in the ICH 150 connects through the token bus 180 to the ICH 150 and ensures that when commanded to prove the state of the isolated execution, the corresponding token (e.g., the motherboard token 182, the token 186) signs only valid isolated digest information. For purposes of security, the token should be connected to the digest memory via the token bus 180.

A HIERARCHICAL EXECUTIVE ARCHITECTURE TO MANAGE A SECURE PLATFORM

The overall architecture discussed above provides a basic insight into a hierarchical executive architecture to manage a secure platform. The elements shown in Figures 1A, 1B, and 1C are instances of an abstract model of this hierarchical executive architecture. The implementation of this hierarchical executive architecture is a combination of hardware and software. In what follows, the processor executive, the processor executive handler, and the operating system executive are abstract models of the processor nub 18, the processor nub loader 52, and the operating system nub 16 (Figures 1A, 1B, and 1C), respectively.

Figure 2 is a diagram illustrating an executive subsystem 200 according to one embodiment of the invention. The executive subsystem 200 includes a processor executive (PE) 210, a PE supplement 220, a PE handler 230, a boot-up code 240, and a secure environment 250.

The processor executive (PE) 210 handles an operating system executive (OSE) 270 in the secure environment 250. The PE supplement 220 supplements the PE with a PE manifest 222 representing the PE and a PE identifier 224 to identify the PE. The PE handler 230 handles the PE 210 using a platform key (PK) 260 in the secure environment 250 and the PE supplement 220. The PE 210 and the PE supplement 220 are located in a PE memory 215. The PE memory 215 is located in the non-isolated memory area 80.

The PE handler 230 handles the PE 210 using the PK 260 and the PE supplement 220. The PE handler 230 obtains information to locate the PE memory 215 via a parameter block 242 provided by the boot-up code 240.

The boot-up code 240 boots up the platform following a power on. The boot-up code 240 obtains an original PE 246 and an original PE supplement 248 from a system ROM (e.g., system flash 160 as shown in Figure 1C)

The secure environment 250 includes a platform key (PK) 260, an operating system executive (OSE) 270, and an OSE supplement 280. The OSE supplement 280 supplements the OSE 270 with an OSE manifest 282 representing the OSE and an OSE identifier 284 to identify the OSE. The secure environment 250 is associated with an isolated memory area 70 (Figure 1C) in the platform. The OSE 270 manages a subset 295 of an operating system (OS) 290 running on the platform. The platform has a processor 110 operating in one of a normal execution mode 112 and an isolated execution mode 115 as shown in Figure 1C. The isolated memory area 70 is accessible to the processor 110 in the isolated execution mode 115.

Figure 3 is a diagram illustrating the PE handler 230 shown in Figure 2 according to one embodiment of the invention. The PE handler 230 includes a PE loader 310, a PE manifest verifier 320, a PE verifier 330, a PE Error Generator 340, a Constant Driver 350, a PE key generator 360, a PE identifier logger 370, and a PE entrance/exit handler 380.

The PE loader 310 loads the PE 210 and the PE supplement 220 from the PE memory 215 (Figure 2) into the isolated memory area 70 using a PE address in the parameter block 242 (Figure 2) provided by the boot-up code 240. The PE loader 310 provides a loaded PE manifest 322 and a loaded PE 312 located in the isolated memory area 70 and corresponding to the PE manifest 322 and the PE 312, respectively.

The PE manifest verifier 320 verifies the PE manifest 222 by comparing the PE manifest 222 with the loaded PE manifest 322 and generates a result to a PE error generator 340. If the verification fails, the error generator 340 generates a failure or fault condition with an error code associated with the PE manifest verification.

The PE verifier 330 verifies the PE 210 using the verified loaded PE manifest 322 and a constant 355 derived from the PK 260 by a constant deriver 350. Essentially, the PE verifier 330 compares the PE 210 with the loaded PE 312. In addition, the PE verifier 330 determines a manifest of the loaded PE 312 using the constant 355 and compares the determined PE manifest with the verified loaded PE manifest 322. The PE verifier 330 then generates a result to the PE error generator 340. If the verification fails, the error generator 340 generates a failure or fault condition with an error code associated with the PE verification.

The PE key generator 360 generates a PE key 365 using the PK 260. The PE key generator 360 includes a PE key combiner 364 to combine the PE identifier 224

and the PK 260. The combined PE identifier 224 and the PK 260 correspond to the PE key 365.

The PE identifier logger 370 logs the PE identifier 224 in a storage 375. The PE identifier logger 370 writes the PE identifier 224 into the storage 375. The storage 375 is a register located inside a chipset such as the ICH 150 shown in Figure 1C.

The PE entrance/exit handler 380 handles a PE entrance and a PE exit. The PE entrance includes obtaining the entry point in the configuration buffer of the processor 110 to represent the PE's entry handler. The PE exit returns control to the boo-up code 240.

Figure 4 is a diagram illustrating the PE 210 shown in Figure 2 according to one embodiment of the invention. The PE 210 includes an OSE loader 410, an OSE manifest verifier 420, an OSE verifier 430, an OSE Error Generator 440, an OSE key generator 460, an OSE identifier logger 470, and an OSE entrance/exit handler 480.

The OSE loader 410loads the OSE 270 and the OSE supplement 280 into the isolated memory area 70 as shown in Figure 2 using an OSE parameter block 405 provided by the OS 290. The OSE loader 410 provides a loaded OSE manifest 422 and a loaded OSE 412 located in the isolated memory area 70 and corresponding to the OSE manifest 282 and the OSE 270, respectively.

The OSE manifest verifier 420 verifies the OSE manifest 282 by comparing the OSE manifest 282 with the loaded OSE manifest 422. The OSE manifest verifier 420 generates a result to an OSE error generator 440. If the verification fails, the OSE error generator 440 generates a failure or fault condition with an error code associated with the OSE manifest verification.

The OSE verifier 430 verifies the OSE 270. Essentially, the OSE verifier 430 compares the OSE 270 with the loaded OSE 412. In addition, the OSE verifier 430 determines a manifest of the loaded OSE 412 using a root key and compares the determined OSE manifest with the verified loaded OSE manifest 422. The OSE verifier 430 then generates a result to the OSE error generator 440. If the verification fails, the OSE error generator 440 generates a failure or fault condition with an error code associated with the OSE verification.

The OSE key generator 460 generates an OSE key 465. The OSE key generator 460 includes a binding key (BK) generator 462 and an OSE key combiner 464. The binding key generator 462 generates a binding key (BK) 463 using the PE key 365 (Figure 3). The OSE key combiner 464 combines the OSE identifier 284 and the BK

463. The combined OSE identifier 284 and the BK 463 correspond to the OSE key 465.

The OSE identifier logger 470 logs the OSE identifier 284 in the storage 375. The storage 375 is a register located inside a chipset such as the ICH 150 shown in Figure 1C.

The OSE entrance/exit handler 480 handles an OSE entrance and an OSE exit. The OSE entrance initializes parameters in a frame buffer and saves appropriate control parameters and transfers control to an entrance handler. The OSE exit clears and creates appropriate return parameters and then transfers control to the exit handler,

Figure 5 is a diagram illustrating the OSE 270 shown in Figure 2 according to one embodiment of the invention. The OSE 270 includes a module loader and evictor 510, a page manager 520, an interface handler 530, a key binder and unbinder 540, a scheduler and balancer 550, and an interrupt handler 560.

The module loader and evictor 510 loads and evicts a module into and out of the isolated memory area 70, respectively. The module is one of an application module 512, an applet module 514, and a support module 516. The page manager 520 manages paging in the isolated memory area 70. The interface handler 530 handles interface with the subset 295 in the OS 290 (Figure 2). The key binder and unbinder 540 includes an applet key generator 542 to generate an applet key 545 associated with the applet module 514. The applet key generator 542 includes an applet key combiner 544 combines the OSE key 465 (Figure 4) with an applet identifier 518 identifying the applet module 514. The combined OSE key 465 and the applet identifier 518 correspond to the applet key 545.

The scheduler and balancer 550 schedules execution of the loaded modules and balances the load of the isolated execution mode. The interrupt handler 560 handles interrupts and exceptions generated in the isolated execution mode.

Figure 6 is a diagram illustrating a boot-up code shown in Figure 2 according to one embodiment of the invention. The boot up code includes a PE locator 610, a PE recorder 620, and an instruction invoker 630.

The PE locator 610 locates the original PE 246 and the original PE supplement 248. The PE locator 610 transfers the original PE 246 and the original PE supplement 248 into the PE memory 215 at a PE address 625. The PE recorder 620 records the PE address 625 in the PE parameter block 242. As discussed above, the PE handler 230

obtains the PE address 625 from the PE parameter block 242 to locate the PE 210 and the PE supplement 220 in the PE memory 215.

The instruction invoker 630 invokes and executes an isolated create instruction 632 which loads the PE handler 230 into the isolated memory area 70. The isolated create instruction 632 performs an atomic non-interruptiblle sequence 640. The atomic sequence 640 includes a number of operations: a physical memory operation 652, an atomic read-and-increment operation 654, an isolated memory area control operation 656, a processor isolated execution operation 658, an PE handler loading operation 663, a PE handler verification 664, and an exit operation 666.

The physical memory operation 652 verifies if the processor is in a flat physical page mode. The atomic read-and-increment operation 654 reads and increments a thread count register in a chipset. The read-and-increment operation 654 determines if the processor is the first processor in the isolated execution mode. The isolated memory area control operation 656 configures the chipset using a configuration storage. The processor isolated execution operation 658 configures the processor in the isolated execution mode. The processor isolated execution operation 658 includes a chipset read operation 672 and a processor configuration operation 674. The chipset read operation 672 reads the configuration storage in the chipset when the processor is not a first processor in the isolated execution mode. The processor configuration operation 674 configures the processor according to the configuration storage read by the chipset read operation 672 when the processor is not a first processor in the isolated execution mode. The PE handler loading operation 662 loads the PE handler 230 into the isolated memory area 70. The PE handler verification 664 verifies the loaded PE handler. The exit operation 666 transfers control to the loaded PE handler.

Figure 7 is a flowchart illustrating a process 700 to manage a secure platform according to one embodiment of the invention.

Upon START, the process 700 boots up the platform following power on (Block 710). The platform has a secure environment. The secure environment includes a platform key, an operating system executive (OSE), and an OSE supplement. The details of the Block 710 are shown in Figure 8. Then, the process 700 handles a processor executive (PE) using the platform key and the PE supplement (Block 720). The details of the Block 720 are shown in Figure 10. Then, the process 700 handles the OSE in the secure environment (Block 730). The details of the Block 730 are shown in Figure 11.

Next, the process 700 manages a subset of an operating system running on the platform (Block 740). The process 700 is then terminated.

Figure 8 is a flowchart illustrating the process 710 to boot up platform according to one embodiment of the invention.

Upon START, the process 710 locates the PE and the PE supplement (Block 810). Then, the process 710 transfers the PE and the PE supplement into the PE memory at a PE address (Block 820). Next, the process 710 records the PE address in a PE parameter block (Block 830). Then, the process 710 executes the isolated create instruction (Block 840). The details of the Block 840 are shown in Figure 9. The process 710 is then terminated.

Figure 9 is a flowchart illustrating the process 840 to execute an isolated create instruction according to one embodiment of the invention.

Upon START, the process 840 determines if the processor is in a flat physical page mode (Block 910). If not, the process 840 sets the processor in the flat physical page mode (Block 915) and proceeds to Block 920. Otherwise, the process 840 determines if the thread count register is zero (Block 920). This is done by reading the thread count register in the chipset to determine if the processor is the first processor in the isolated execution mode. If not, the process 840 determines that the processor is not the first processor in the system to be in the isolated execution mode. The process 840 then reads the configuration storage from the chipset (Block 925). Then, the process 840 configured the processor using the chipset configuration storage (Block 930). Then, the process 840 proceeds to Block 960.

If the thread count register is zero, the process 840 determines that the processor is the first processor in the system to be booted up with isolated execution mode. The process 840 then increments the thread count register to inform to other processors that there is already a processor being booted up in isolated execution mode (Block 935). Then, the process 840 configures the chipset and the processor in isolated execution mode by writing appropriate setting values (e.g., isolated mask and base values) in the chipset and processor configuration storage (Block 940). To configure the processor, the process 840 may also need to set up the isolated execution mode word in the control register of the processor.

Next, the process 840 loads the PE handler from the ROM internal to the chipset to the isolated memory area (Block 945). Then, the process 840 determines if the loaded PE handler is the same as the original PE handler in the ROM (Block 950). If

not, the process 840 generates a failure or fault condition with an appropriate error code (Block 955) and is then terminated. Otherwise, the process 840 transfers control to the loaded PE handler (Block 960). The process 840 is then terminated.

Figure 10 is a flowchart illustrating the process 720 to handle a processor executive according to one embodiment of the invention.

Upon START, the process 720 loads the PE and the PE supplement from a PE memory into the isolated memory area using a parameter block provided by the boot-up code (Block 1010). Next, the process 720 determines if the loaded PE manifest is the same as the original PE manifest (Block 1015). If not, the process 720 generates a failure or fault condition with appropriate error code (Block 1020) and is then terminated. Otherwise, the process 720 determines if the loaded PE has the same manifest as the loaded PE manifest (Block 1025). If not, the process 720 goes to Block 1020 and is then terminated. Otherwise, the process 720 generates a PE key using the platform key in the secure environment (Block 1030).

Then, the process 720 logs the PE identifier in a storage (Block 1035). This log storage is typically a register in an ICH. Then, the process 720 changes the entry point in the configuration buffer of the processor to prepare for an OSE entrance (Block 1040). Then, the process 720 returns to the boot-up code (Block 1045). The process 720 is then terminated.

Figure 11 is a flowchart illustrating the process 730 to handle the OSE according to one embodiment of the invention.

Upon START, the OS boots and locates the OSE and the OSE supplement in the OSE memory at an OSE address (Block 1110). Then the OS records the OSE address in an OSE parameter block (Block 1115). Next, the process 730 determines if an OSE has already been loaded (Block 1120). If yes, the process 730 is terminated. Otherwise, the process 730 loads the OSE and the OSE supplement into the isolated memory area (Block 1125).

Next, the process 730 determines if the loaded OSE manifest is the same as the original OSE manifest (Block 1130). If not, the process 730 generates a failure or fault condition with an appropriate error code (Block 1135) and is then terminated. Otherwise, the process 730 determines if the loaded OSE has the same manifest as the loaded OSE manifest (Block 1140). If not, the process 730 goes to block 1135 and is then terminated. Otherwise, the process 730 generates the OSE key using the PE key and the OSE identifier (Block 1145).

Then, the process 730 logs the OSE identifier in a storage (Block 1150). Typically, this log storage is a register in a chipset such as the ICH. Next, the process 730 clears any PE secrets or services that are not needed (Block 1155). Then, the process 730 returns to the PE's exit handler (Block 1160). The process 730 is then terminated.

While this invention has been described with reference to illustrative embodiments, this description is not intended to be construed in a limiting sense. Various modifications of the illustrative embodiments, as well as other embodiments of the invention, which are apparent to persons skilled in the art to which the invention pertains are deemed to lie within the spirit and scope of the invention.

CLAIMS

What is claimed is:

1. An apparatus comprising:

a processor executive (PE) to handle an operating system executive (OSE) in a secure environment, the secure environment having a platform key (PK) and associated with an isolated memory area in a platform, the OSE to manage a subset of an operating system (OS) running on the platform, the platform having a processor operating in one of a normal execution mode and an isolated execution mode, the isolated memory area being accessible to the processor in the isolated execution mode;

a PE supplement to supplement the PE with a PE manifest representing the PE and a PE identifier to identify the PE; and

a PE handler to handle the PE using the PK and the PE supplement.

2. The apparatus of claim 1 further comprises:

a boot-up code to boot up the platform following a power on.

3. The apparatus of claim 2 wherein the secure environment includes an OSE supplement to supplement the OSE with an OSE manifest representing the OSE and an OSE identifier to identify the OSE.

4. The apparatus of claim 3 wherein the PE handler comprises:

a PE loader to load the PE and the PE supplement from a PE memory into the isolated memory area using a parameter block provided by the boot-up code;

a PE manifest verifier to verify the PE manifest; and

a PE verifier to verify the PE using the PE manifest and a constant derived from the PK.

5. The apparatus of claim 4 wherein the PE handler further comprises:

a PE key generator to generate a PE key using the PK;

a PE identifier logger to log the PE identifier in a storage; and

a PE entrance/exit handler to handle a PE entry and a PE exit.

6. The apparatus of claim 5 wherein the PE key generator comprises:

a PE key combiner to combine the PE identifier and the PK, the combined PE identifier and the PK corresponding to the PE key.

7. The apparatus of claim 6 wherein the PE comprises:

an OSE loader to load the OSE and the OSE supplement into the isolated memory area;

an OSE manifest verifier to verify the OSE manifest; and

an OSE verifier to verify the OSE.

8. The apparatus of claim 7 wherein the PE further comprises:

an OSE key generator to generate an OSE key;

an OSE identifier logger to log the OSE identifier in a storage; and

an OSE entrance/exit handler to handle an OSE entry and an OSE exit.

9. The apparatus of claim 8 wherein the OSE key generator comprises:

a binding key generator to generate a binding key (BK) using the PE key; and

an OSE key combiner to combine the OSE identifier and the BK, the combined OSE identifier and the BK corresponding to the OSE key.

10. The apparatus of claim 9 wherein the OSE comprises:

a module loader to load a module into the isolated memory area;

a page manager to manage paging in the isolated memory area; and

an interface handler to handle interface with the OS.

11. The apparatus of claim 9 wherein the module is one of an application module, an applet module, and a support module.

12. The apparatus of claim 11 wherein the OSE further comprises:

an applet key generator to generate an applet key associating with the applet module.

13. The apparatus of claim 12 wherein the applet key generator comprises:

an applet key combiner to combine the OSE key with an applet identifier identifying the applet module, the combined OSE key and the applet identifier corresponding to the applet key.

14. The apparatus of claim 13 wherein the boot up code comprises:

a PE locator to locate the PE and the PE supplement, the PE locator transferring the PE and the PE supplement into the PE memory at a PE address;

a PE recorder to record the PE address in the parameter block; and

an instruction invoker to execute an isolated create instruction, the isolated create instruction loading the PE handler into the isolated memory area.

15. The apparatus of claim 14 wherein the isolated create instruction performs an atomic sequence, the atomic sequence being non-interruptible.

16. The apparatus of claim 15 wherein the atomic sequence comprises:

a physical memory operation to verify if the processor is in a flat physical page mode;

an atomic read-and-increment operation to read and increment a thread count register in a chipset, the read-and-increment operation determining if the processor is the first processor in the isolated execution mode;

an isolated memory area control operation to configure the chipset using a configuration storage;

a processor isolated execution operation to configure the processor in the isolated execution mode; and

an PE handler loading operation to load the PE handler into the isolated memory area.

17. The apparatus of claim 16 wherein the atomic sequence further comprises:

a PE handler verification to verify the loaded PE handler; and

an exit operation to transfer control to the loaded PE handler.

18. The apparatus of claim 16 wherein the processor isolated execution operation comprises:

a chipset read operation to read the configuration storage in the chipset when the processor is not a first processor in the isolated execution mode; and

a processor configuration operation to configure the processor according to the configuration storage in the chipset when the processor is not the first processor in the isolated execution mode.

19. The apparatus of claim 18 wherein the chipset includes at least one of a memory controller hub (MCH) and an input/output controller hub (ICH).

20. The apparatus of claim 8 wherein the storage is in an input/output controller hub (ICH) external to the processor.

21. A method comprising:

handling an operating system executive (OSE) by a processor executive (PE) in a secure environment, the secure environment having a platform key (PK) and associated with an isolated memory area in a platform, the OSE to manage a subset of an operating system (OS) running on the platform, the platform having a processor operating in one of a normal execution mode and an isolated execution mode, the isolated memory area being accessible to the processor in the isolated execution mode;

supplementing the PE using a PE supplement, the PE supplement having a PE manifest representing the PE and a PE identifier to identify the PE; and

handling the PE by a PE handler using the PK and the PE supplement.

22. The method of claim 21 further comprises:

booting up the platform by a boot-up code following a power on.

23. The method of claim 22 wherein the secure environment includes an OSE supplement to supplement the OSE with an OSE manifest representing the OSE and an OSE identifier to identify the OSE.

24. The method of claim 23 wherein handling the PE comprises:

loading the PE and the PE supplement from a PE memory into the isolated memory area using a parameter block provided by the boot-up code;

verifying the PE manifest; and

verifying the PE using the PE manifest and a constant derived from the PK.

25. The method of claim 24 wherein handling the PE further comprises:

generating a PE key using the PK;

logging the PE identifier in a storage; and

handling a PE entry and a PE exit.

26. The method of claim 25 wherein generating the PE key comprises:

combining the PE identifier and the PK, the combined PE identifier and the PK corresponding to the PE key.

27. The method of claim 26 wherein handling the OSE comprises:

loading the OSE and the OSE supplement into the isolated memory area;

verifying the OSE manifest; and

verifying the OSE.

28. The method of claim 27 wherein handling the OSE further comprises:

generating an OSE key;

logging the OSE identifier in a storage; and

handling an OSE entry and an OSE exit.

29. The method of claim 28 wherein generating the OSE key comprises:

generating a binding key (BK) using the PE key; and

combining the OSE identifier and the BK, the combined OSE identifier and the BK corresponding to the OSE key.

30. The method of claim 29 wherein managing the subset of the OS comprises:

loading a module into the isolated memory area;
managing paging in the isolated memory area; and
handling interface with the OS.

31. The method of claim 29 wherein the module is one of an application module, an applet module, and a support module.

32. The method of claim 31 wherein managing the subset of the OS further comprises:
generating an applet key associating with the applet module.

33. The method of claim 32 wherein generating the applet key comprises:
combining the OSE key with an applet identifier identifying the applet module, the combined OSE key and the applet identifier corresponding to the applet key.

34. The method of claim 33 wherein booting up comprises:
locating the PE and the PE supplement;
transferring the PE and the PE supplement into the PE memory at a PE address;
recording the PE address in the parameter block; and
executing an isolated create instruction, the isolated create instruction loading the PE handler into the isolated memory area.

35. The method of claim 34 wherein executing the isolated create instruction comprises performing an atomic sequence, the atomic sequence being non-interruptible.

36. The method of claim 35 wherein performing the atomic sequence comprises:
verifying if the processor is in a flat physical page mode;
reading and incrementing a thread count register in a chipset to determine if the processor is the first processor in the isolated execution mode;
configuring the chipset using a configuration storage;
configuring the processor in the isolated execution mode; and
loading the PE handler into the isolated memory area.

37. The method of claim 36 wherein performing the atomic sequence further comprises:

verifying the loaded PE handler; and

transferring control to the loaded PE handler.

38. The method of claim 36 wherein configuring the processor in the isolated execution mode comprises:

reading the configuration storage in the chipset when the processor is not a first processor in the isolated execution mode; and

configuring the processor according to the configuration storage in the chipset when the processor is not the first processor in the isolated execution mode.

39. The method of claim 38 wherein the chipset includes at least one of a memory controller hub (MCH) and an input/output controller hub (ICH).

40. The method of claim 28 wherein the storage is in an input/output controller hub (ICH) external to the processor.

41. A computer program product comprising:

a machine useable medium having computer program code embedded therein, the computer program product having:

computer readable program code for handling an operating system executive (OSE) by a processor executive (PE) in a secure environment, the secure environment having a platform key (PK) and associated with an isolated memory area in a platform, the OSE to manage a subset of an operating system (OS) running on the platform, the platform having a processor operating in one of a normal execution mode and an isolated execution mode, the isolated memory area being accessible to the processor in the isolated execution mode;

computer readable program code for supplementing the PE using a PE supplement, the PE supplement having a PE manifest representing the PE and a PE identifier to identify the PE; and

computer readable program code for handling the PE by a PE handler using the PK and the PE supplement.

42. The computer program product of claim 41 further comprises:

computer readable program code for booting up the platform by a boot-up code following a power on.

43. The computer program product of claim 42 wherein the secure environment includes an OSE supplement to supplement the OSE with an OSE manifest representing the OSE and an OSE identifier to identify the OSE.

44. The computer program product of claim 43 wherein the computer readable program code for handling the PE comprises:

computer readable program code for loading the PE and the PE supplement from a PE memory into the isolated memory area using a parameter block provided by the boot-up code;

computer readable program code for verifying the PE manifest; and

computer readable program code for verifying the PE using the PE manifest and a constant derived from the PK.

45. The computer program product of claim 44 wherein the computer readable program code for handling the PE further comprises:

computer readable program code for generating a PE key using the PK;

computer readable program code for logging the PE identifier in a storage; and

computer readable program code for handling a PE entry and a PE exit.

46. The computer program product of claim 45 wherein the computer readable program code for generating the PE key comprises:

computer readable program code for combining the PE identifier and the , the combined PE identifier and the PK corresponding to the PE key.

47. The computer program product of claim 46 wherein the computer readable program code for handling the OSE comprises:

computer readable program code for loading the OSE and the OSE supplement into the isolated memory area;

computer readable program code for verifying the OSE manifest; and

computer readable program code for verifying the OSE.

48. The computer program product of claim 47 wherein the computer readable program code for handling the OSE further comprises:

computer readable program code for generating an OSE key;

computer readable program code for logging the OSE identifier in a storage; and

computer readable program code for handling an OSE entry and an OSE exit.

49. The computer program product of claim 48 wherein the computer readable program code for generating the OSE key comprises:

computer readable program code for generating a binding key (BK) using the PE key; and

computer readable program code for combining the OSE identifier and the BK, the combined OSE identifier and the BK corresponding to the OSE key.

50. The computer program product of claim 49 wherein the computer readable program code for managing the subset of the OS comprises:

computer readable program code for loading a module into the isolated memory area;

computer readable program code for managing paging in the isolated memory area; and

computer readable program code for handling interface with the OS.

51. The computer program product of claim 49 wherein the module is one of an application module, an applet module, and a support module.

52. The computer program product of claim 51 wherein the computer readable program code for managing the subset of the OS further comprises:

computer readable program code for generating an applet key associating with the applet module.

53. The computer program product of claim 52 wherein the computer readable program code for generating the applet key comprises:

computer readable program code for combining the OSE key with an applet identifier identifying the applet module, the combined OSE key and the applet identifier corresponding to the applet key.

54. The computer program product of claim 53 wherein the computer readable program code for booting up comprises:

computer readable program code for locating the PE and the PE supplement;

computer readable program code for transferring the PE and the PE supplement into the PE memory at a PE address;

computer readable program code for recording the PE address in the parameter block; and

computer readable program code for executing an isolated create instruction, the isolated create instruction loading the PE handler into the isolated memory area.

55. The computer program product of claim 54 wherein the computer readable program code for executing the isolated create instruction comprises computer readable program code for performing an atomic sequence, the atomic sequence being non-interruptible.

56. The computer program product of claim 55 wherein the computer readable program code for performing the atomic sequence comprises:

computer readable program code for verifying if the processor is in a flat physical page mode;

computer readable program code for reading and incrementing a thread count register in a chipset to determine if the processor is the first processor in the isolated execution mode;

computer readable program code for configuring the chipset using a configuration storage;

computer readable program code for configuring the processor in the isolated execution mode; and

computer readable program code for loading the PE handler into the isolated memory area.

57. The computer program product of claim 56 wherein the computer readable program code for performing the atomic sequence further comprises:

computer readable program code for verifying the loaded PE handler; and

computer readable program code for transferring control to the loaded PE handler.

58. The computer program product of claim 56 wherein the computer readable program code for configuring the processor in the isolated execution mode comprises:

computer readable program code for reading the configuration storage in the chipset when the processor is not a first processor in the isolated execution mode; and

computer readable program code for configuring the processor according to the configuration storage in the chipset when the processor is not the first processor in the isolated execution mode.

59. The computer program product of claim 58 wherein the chipset includes at least one of a memory controller hub (MCH) and an input/output controller hub (ICH).

60. The computer program product of claim 48 wherein the storage is in an input/output controller hub (ICH) external to the processor.

61. A system comprising:

a processor operating in one of a normal execution mode and an isolated execution mode;

a memory coupled to the processor having an isolated memory area accessible to the processor in the isolated execution mode; and

an executive subsystem comprising:

a processor executive (PE) to handle an operating system executive (OSE) in a secure environment, the secure environment having a platform key

(PK) and associated with the isolated memory, the OSE to manage a subset of an operating system (OS),

a PE supplement to supplement the PE with a PE manifest representing the PE and a PE identifier to identify the PE, and

a PE handler to handle the PE using the PK and the PE supplement.

62. The system of claim 61 wherein the executive subsystem further comprises:

a boot-up code to boot up the platform following a power on.

63. The system of claim 62 wherein the secure environment includes an OSE supplement to supplement the OSE with an OSE manifest representing the OSE and an OSE identifier to identify the OSE.

64. The system of claim 63 wherein the PE handler comprises:

a PE loader to load the PE and the PE supplement from a PE memory into the isolated memory area using a parameter block provided by the boot-up code;

a PE manifest verifier to verify the PE manifest; and

a PE verifier to verify the PE using the PE manifest and a constant derived from the PK.

65. The system of claim 64 wherein the PE handler further comprises:

a PE key generator to generate a PE key using the PK;

a PE identifier logger to log the PE identifier in a storage; and

a PE entrance/exit handler to handle a PE entry and a PE exit.

66. The system of claim 65 wherein the PE key generator comprises:

a PE key combiner to combine the PE identifier and the PK, the combined PE identifier and the PK corresponding to the PE key.

67. The system of claim 66 wherein the PE comprises:

an OSE loader to load the OSE and the OSE supplement into the isolated memory area;

an OSE manifest verifier to verify the OSE manifest; and
an OSE verifier to verify the OSE.

68. The system of claim 67 wherein the PE further comprises:
an OSE key generator to generate an OSE key;
an OSE identifier logger to log the OSE identifier in a storage; and
an OSE entrance/exit handler to handle an OSE entry and an OSE exit.

69. The system of claim 68 wherein the OSE key generator comprises:
a binding key generator to generate a binding key (BK) using the PE key; and
an OSE key combiner to combine the OSE identifier and the BK, the combined OSE identifier and the BK corresponding to the OSE key.

70. The system of claim 69 wherein the OSE comprises:
a module loader to load a module into the isolated memory area;
a page manager to manage paging in the isolated memory area; and
an interface handler to handle interface with the OS.

71. The system of claim 69 wherein the module is one of an application module, an applet module, and a support module.

72. The system of claim 71 wherein the OSE further comprises:
an applet key generator to generate an applet key associating with the applet module.

73. The system of claim 72 wherein the applet key generator comprises:
an applet key combiner to combine the OSE key with an applet identifier identifying the applet module, the combined OSE key and the applet identifier corresponding to the applet key.

74. The system of claim 73 wherein the boot up code comprises:
a PE locator to locate the PE and the PE supplement, the PE locator transferring the PE and the PE supplement into the PE memory at a PE address;

a PE recorder to record the PE address in the parameter block; and

an instruction invoker to execute an isolated create instruction, the isolated create instruction loading the PE handler into the isolated memory area.

75. The system of claim 74 wherein the isolated create instruction performs an atomic sequence, the atomic sequence being non-interruptible.

76. The system of claim 75 wherein the atomic sequence comprises:

a physical memory operation to verify if the processor is in a flat physical page mode;

an atomic read-and-increment operation to read and increment a thread count register in a chipset, the read-and-increment operation determining if the processor is the first processor in the isolated execution mode;

an isolated memory area control operation to configure the chipset using a configuration storage;

a processor isolated execution operation to configure the processor in the isolated execution mode; and

an PE handler loading operation to load the PE handler into the isolated memory area.

77. The system of claim 76 wherein the atomic sequence further comprises:

a PE handler verification to verify the loaded PE handler; and

an exit operation to transfer control to the loaded PE handler.

78. The system of claim 76 wherein the processor isolated execution operation comprises:

a chipset read operation to read the configuration storage in the chipset when the processor is not a first processor in the isolated execution mode; and

a processor configuration operation to configure the processor according to the configuration storage in the chipset when the processor is not the first processor in the isolated execution mode.

79. The system of claim 78 wherein the chipset includes at least one of a memory controller hub (MCH) and an input/output controller hub (ICH).

80. The system of claim 68 wherein the storage is in an input/output controller hub (ICH) external to the processor.

ABSTRACT OF THE DISCLOSURE

A processor executive (PE) handles an operating system executive (OSE) in a secure environment. The secure environment has a platform key (PK) and is associated with an isolated memory area in the platform. The OSE manages a subset of an operating system (OS) running on the platform. The platform has a processor operating in one of a normal execution mode and an isolated execution mode. The isolated memory area is accessible to the processor in the isolated execution mode. A PE supplement supplements the PE with a PE manifest representing the PE and a PE identifier to identify the PE. A PE handler handles the PE using the PK and the PE supplement.

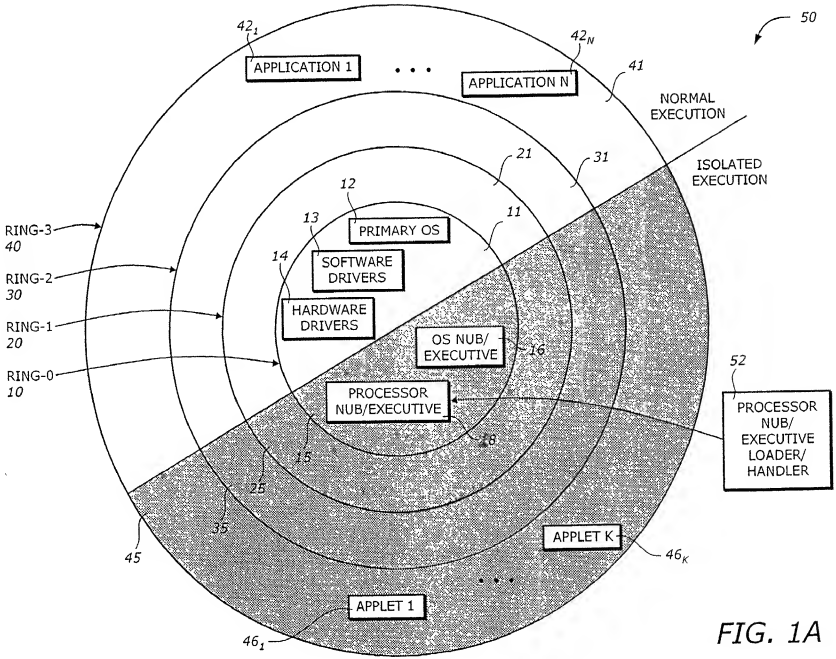


FIG. 1A

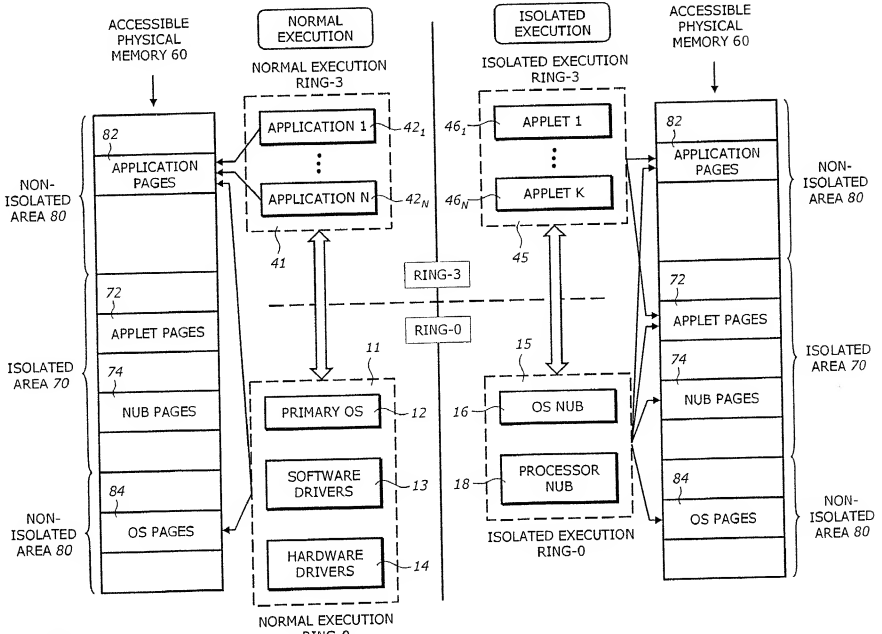


FIG. 1B

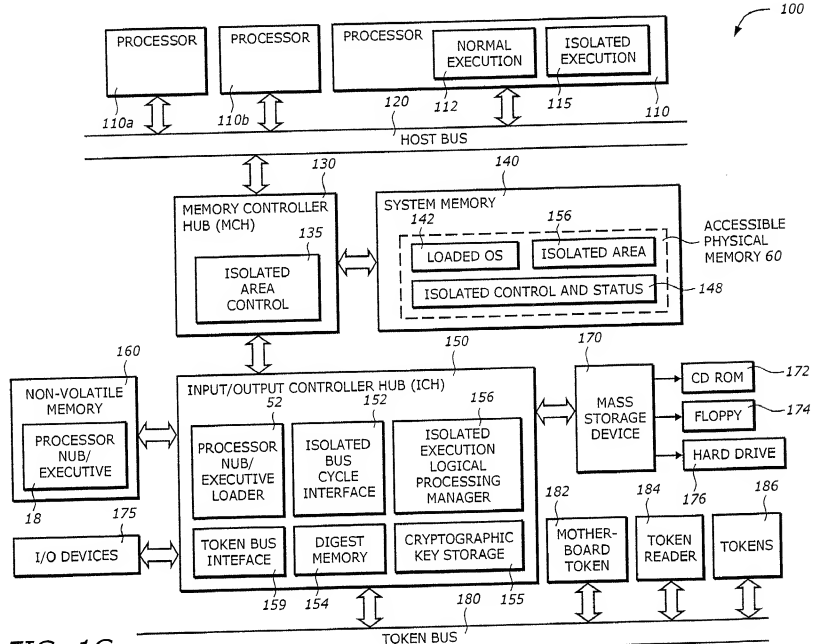


FIG. 1C

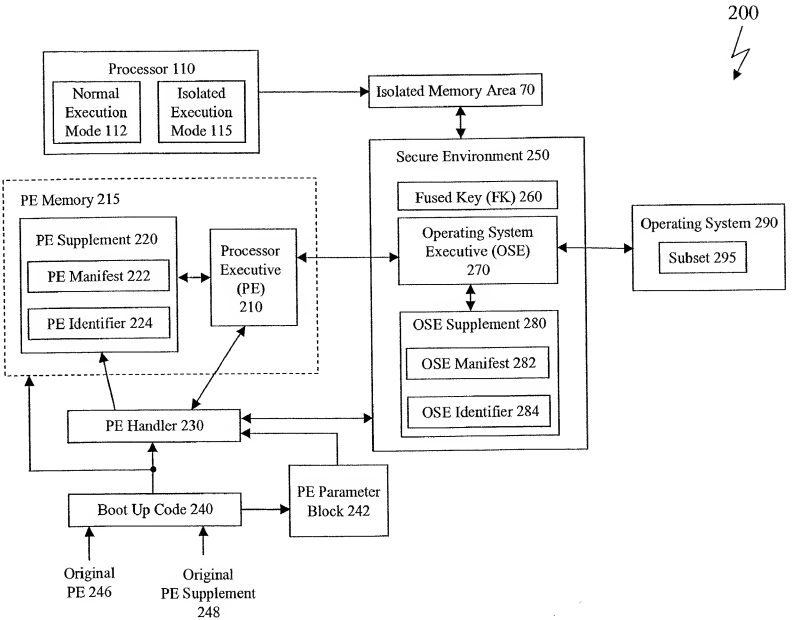


Fig. 2

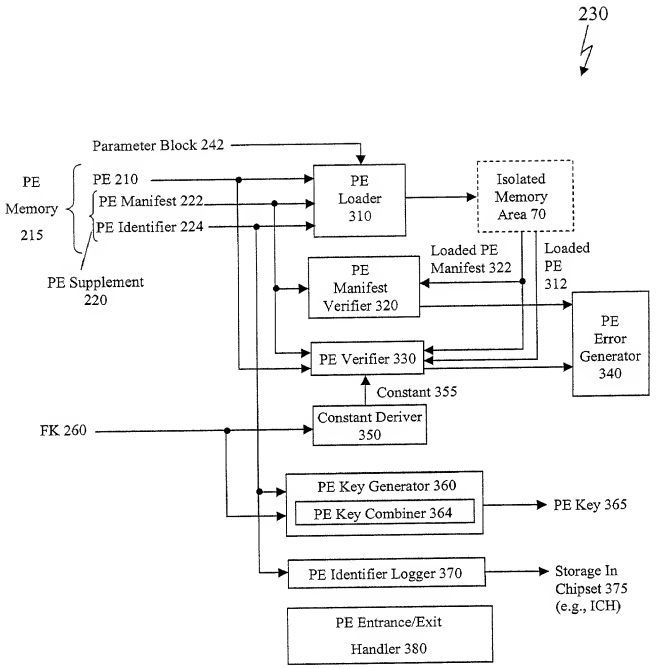


Fig. 3

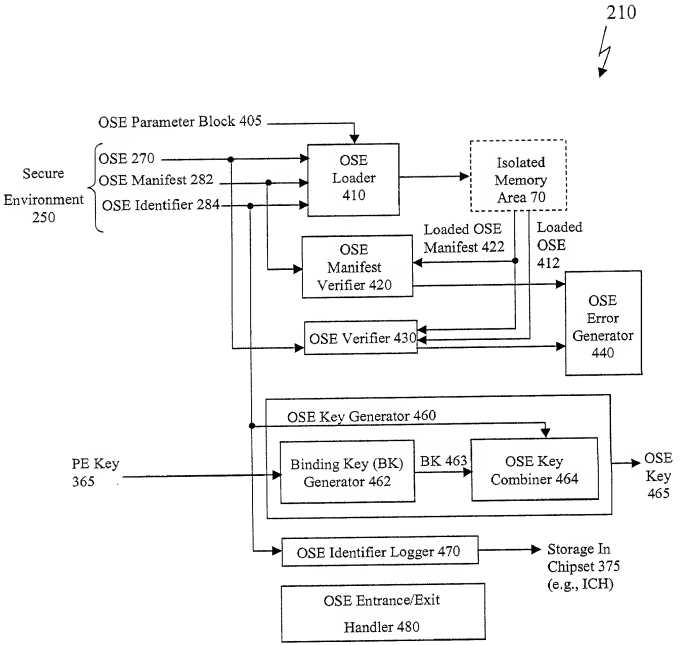


Fig. 4

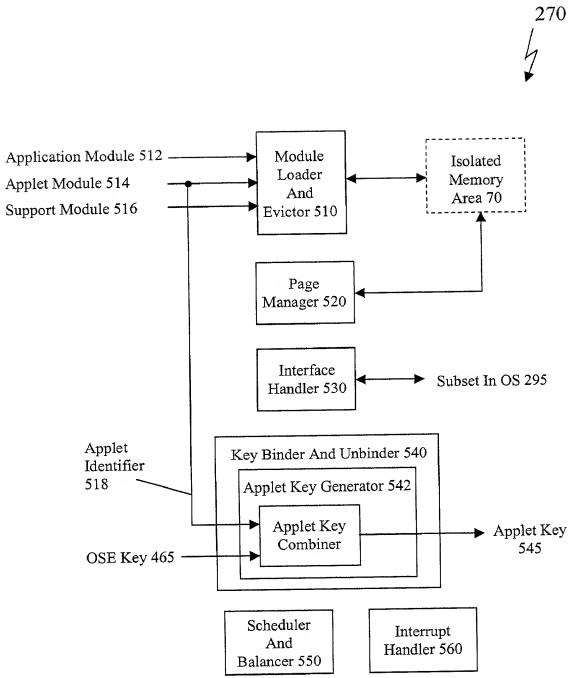

270
Application Module 512
Applet Module 514
Support Module 516
Module Loader And Evictor 510
Isolated Memory Area 70
Page Manager 520
Interface Handler 530
Subset In OS 295
Applet Identifier 518
Key Binder And Unbinder 540
Applet Key Generator 542
Applet Key Combiner
Applet Key 545
OSE Key 465
Scheduler And Balancer 550
Interrupt Handler 560

Fig. 5

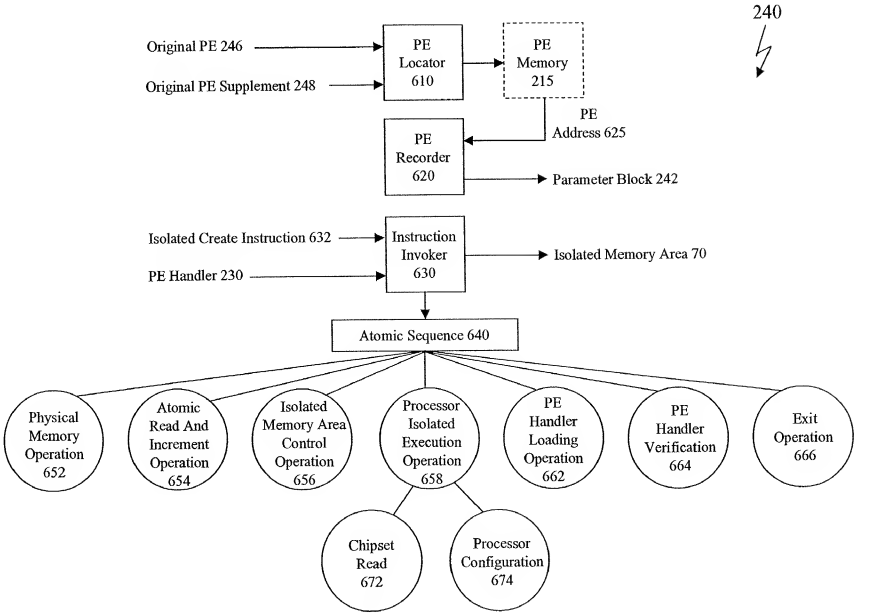


Fig. 6

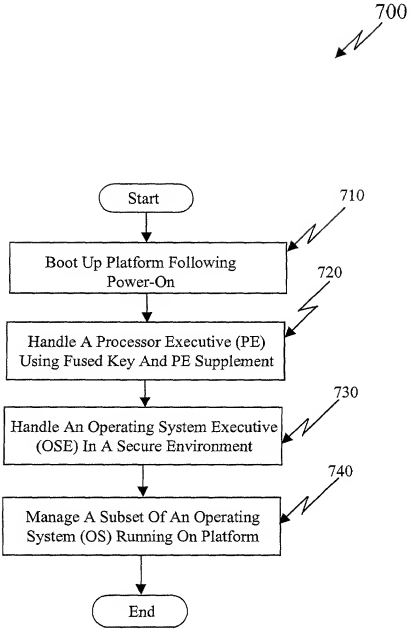


Fig. 7

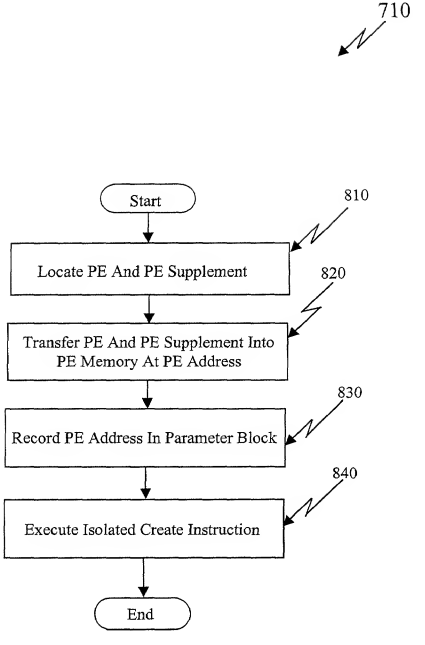


Fig. 8

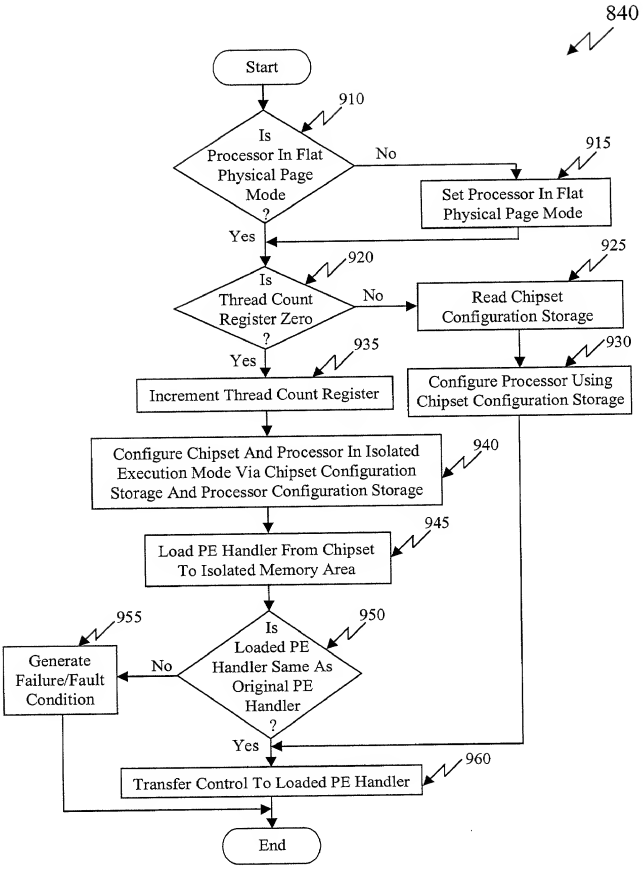

840
Start
910
Is Processor In Flat Physical Page Mode ?
No
915
Set Processor In Flat Physical Page Mode
Yes
920
Is Thread Count Register Zero ?
No
925
Read Chipset Configuration Storage
Yes
935
930
Increment Thread Count Register
Configure Processor Using Chipset Configuration Storage
940
Configure Chipset And Processor In Isolated Execution Mode Via Chipset Configuration Storage And Processor Configuration Storage
945
Load PE Handler From Chipset To Isolated Memory Area
955
950
Generate Failure/Fault Condition
No
Is Loaded PE Handler Same As Original PE Handler ?
Yes
960
Transfer Control To Loaded PE Handler
End

Fig. 9

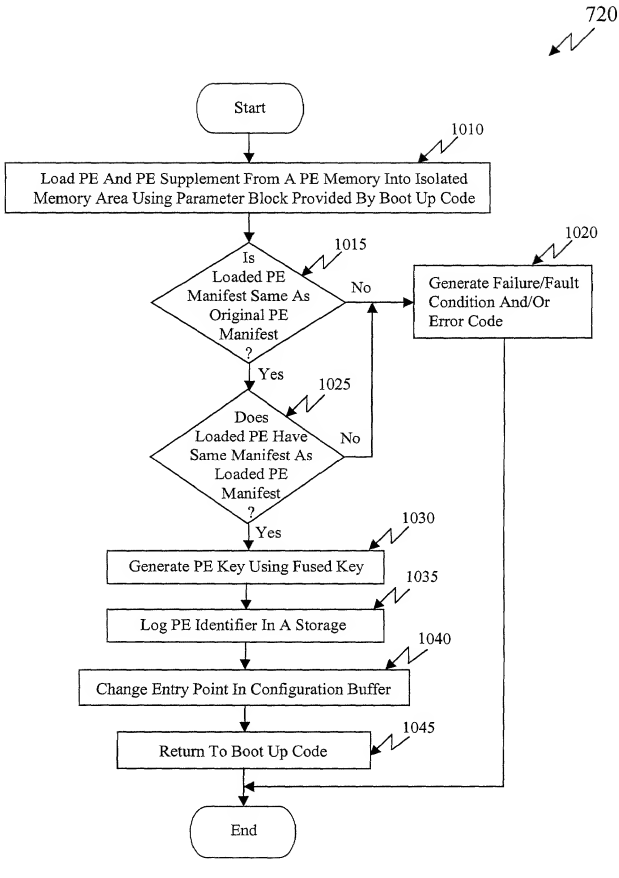


Fig. 10

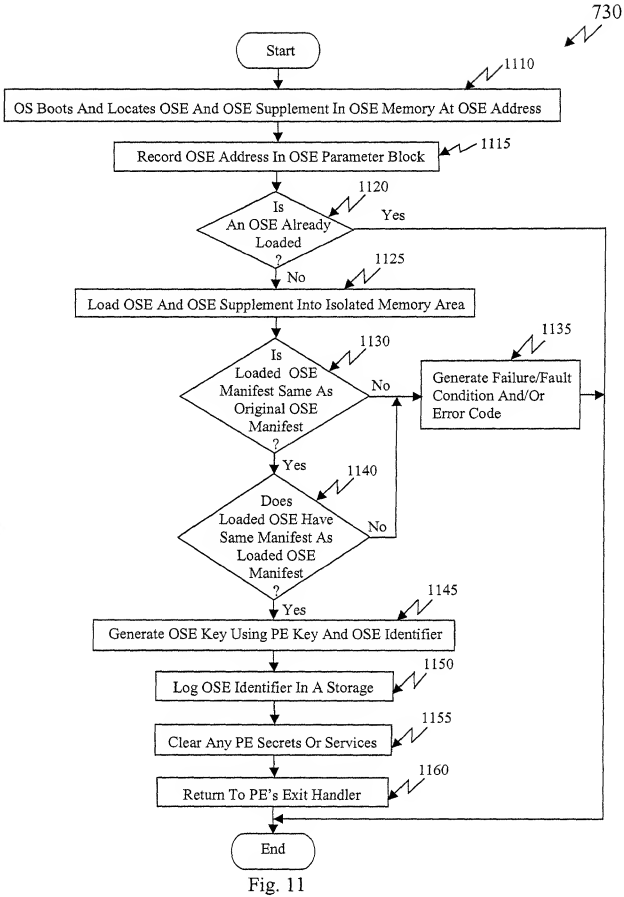

730
Start
1110
OS Boots And Locates OSE And OSE Supplement In OSE Memory At OSE Address
1115
Record OSE Address In OSE Parameter Block
1120
Is An OSE Already Loaded ?
Yes
No
1125
Load OSE And OSE Supplement Into Isolated Memory Area
1130
Is Loaded OSE Manifest Same As Original OSE Manifest ?
No
1135
Generate Failure/Fault Condition And/Or Error Code
Yes
1140
Does Loaded OSE Have Same Manifest As Loaded OSE Manifest ?
No
Yes
1145
Generate OSE Key Using PE Key And OSE Identifier
1150
Log OSE Identifier In A Storage
1155
Clear Any PE Secrets Or Services
1160
Return To PE's Exit Handler
End

Fig. 11

Attorney's Docket No.: 042390.P8098X

Patent

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION (CONTINUATION-IN-PART)

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below, next to my name.

I believe I am the original, first, and sole inventor (if only one name is listed below) or any original, first, and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

MANAGING A SECURE ENVIRONMENT USING A HIERARCHICAL EXECUTIVE ARCHITECTURE IN ISOLATED EXECUTION MODE

the specification of which

[X] is attached hereto.
[ ] was filed on _____ as
United States Application Number _____
or PCT International Application Number _____
and was amended on _____
(if applicable)

That this application in part discloses and claims subject matter disclosed in my earlier filed pending application:

Application No.: 09/539,344
Filed: 3/31/2000

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

That as to the subject matter of this application which is common to said earlier application, I do not know and do not believe that the same was ever known or used in the United States of America before my invention thereof, or patented or described in any printed publication in any country before my invention thereof or more than one year prior to this application, that the same was not in public use or on sale in the United States of America more than one year prior to this application, and

That said common subject matter has not been patented or made the subject of an inventor's certificate issued before the date of said earlier application in any country foreign to the United States of America on an application filed by me or my legal representatives or assigns more than twelve months prior to said earlier application;

I acknowledge the duty to disclose information which is material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119, of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

APPLICATION NUMBER	COUNTRY (OR INDICATE IF PCT)	DATE OF FILING (day, **month**, year)	PRIORITY CLAIMED UNDER 37 USC 119
			☐ No ☐ Yes
			☐ No ☐ Yes
			☐ No ☐ Yes

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose all information known to me to be material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application:

APPLICATION NUMBER	FILING DATE	STATUS (ISSUED, PENDING, ABANDONED)

I hereby appoint the persons listed on Appendix A hereto (which is incorporated by reference and a part of this document) as my respective patent attorneys and patent agents, with full power of substitution and revocation, to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith.

Send correspondence to:
Thinh V. Nguyen, Reg. No. 42,034, BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN, LLP
(Name of Attorney or Agent)
12400 Wilshire Boulevard, 7th Floor, Los Angeles, California 90025 and direct telephone calls to:
Thinh V. Nguyen, (714) 557-3800.
(Name of Attorney or Agent)

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Sole/First Inventor (given name, family name) **Carl M. Ellison**

Inventor's Signature Carl M. Ellison Date 8/18/00

Residence Portland, Oregon USA (*City , State*) Citizenship USA (*Country*)

P. O. Address 1818 N.W. 28th Avenue
Portland, Oregon 97210-2214 USA

Full Name of Second/Joint Inventor (given name, family name) **Roger A. Golliver**

Inventor's Signature _____ Date _____

Residence Beaverton, Oregon USA (*City , State*) Citizenship USA (*Country*)

P. O. Address 16185 SW Night Hawk Drive
Beaverton, Oregon 97007-8368 USA

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose all information known to me to be material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application:

APPLICATION NUMBER	FILING DATE	STATUS (ISSUED, PENDING, ABANDONED)

I hereby appoint the persons listed on Appendix A hereto (which is incorporated by reference and a part of this document) as my respective patent attorneys and patent agents, with full power of substitution and revocation, to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith.

Send correspondence to:
Thinh V. Nguyen, Reg. No. 42,034, BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN, LLP
(Name of Attorney or Agent)
12400 Wilshire Boulevard, 7th Floor, Los Angeles, California 90025 and direct telephone calls to:
Thinh V. Nguyen, (714) 557-3800.
(Name of Attorney or Agent)

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Sole/First Inventor (given name, family name) **Carl M. Ellison**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA (*City , State*) Citizenship USA (*Country*)

P. O. Address 1818 N.W. 28th Avenue
Portland, Oregon 97210-2214 USA

Full Name of Second/Joint Inventor (given name, family name) **Roger A. Golliver**

Inventor's Signature Roger A Golliver Date 7/28/2000

Residence Beaverton, Oregon USA (*City , State*) Citizenship USA (*Country*)

P. O. Address 16185 SW Night Hawk Drive
Beaverton, Oregon 97007-8368 USA

Full Name of Third/Joint Inventor (given name, family name) **Howard C. Herbert**

Inventor's Signature Howard C. Herbert Date 24 July 2000

Residence Phoenix, Arizona USA Citizenship USA
(City , State) *(Country)*

P. O. Address 16817 South 1st Drive
Phoenix, Arizona 85045 USA

Full Name of Fourth/Joint Inventor (given name, family name) **Derrick C. Lin**

Inventor's Signature _____ Date _____

Residence Foster City, California USA Citizenship USA
(City , State) *(Country)*

P. O. Address 113 Barkentine Street
Foster City, California 94404 USA

Full Name of Fifth/Joint Inventor (given name, family name) **Francis X. McKeen**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 10612 N. W. LeMans Ct.
Portland, Oregon 97229 USA

Full Name of Sixth/Joint Inventor (given name, family name) **Gilbert N. Neiger**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 2424 N. E. 11th Avenue
Portland, Oregon 97212 USA

Full Name of Third/Joint Inventor (given name, family name) **Howard C. Herbert**

Inventor's Signature _____ Date _____

Residence Phoenix, Arizona USA Citizenship USA
(City , State) *(Country)*

P. O. Address 16817 South 1st Drive
Phoenix, Arizona 85045 USA

Full Name of Fourth/Joint Inventor (given name, family name) **Derrick C. Lin**

Inventor's Signature  Date 8/30/02

Residence San Mateo, California USA Citizenship USA
(City , State) *(Country)*

P. O. Address 1737 Oakwood Drive
San Mateo, California 9440 USA

Full Name of Fifth/Joint Inventor (given name, family name) **Francis X. McKeen**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 10612 N. W. LeMans Ct.
Portland, Oregon 97229 USA

Full Name of Sixth/Joint Inventor (given name, family name) **Gilbert N. Neiger**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 2424 N. E. 11th Avenue
Portland, Oregon 97212 USA

Full Name of Third/Joint Inventor (given name, family name) **Howard C. Herbert**

Inventor's Signature _____ Date _____

Residence Phoenix, Arizona USA Citizenship USA
(City , State) *(Country)*

P. O. Address 16817 South 1st Drive
Phoenix, Arizona 85045 USA

Full Name of Fourth/Joint Inventor (given name, family name) **Derrick C. Lin**

Inventor's Signature _____ Date _____

Residence Foster City, California USA Citizenship USA
(City , State) *(Country)*

P. O. Address 113 Barkentine Street
Foster City, California 94404 USA

Full Name of Fifth/Joint Inventor (given name, family name) **Francis X. McKeen**

Inventor's Signature Francis X. McKeen Date 8/3/01

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 10612 N. W. LeMans Ct.
Portland, Oregon 97229 USA

Full Name of Sixth/Joint Inventor (given name, family name) **Gilbert N. Neiger**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 2424 N. E. 11th Avenue
Portland, Oregon 97212 USA

Full Name of Third/Joint Inventor (given name, family name) **Howard C. Herbert**

Inventor's Signature _____ Date _____

Residence Phoenix, Arizona USA Citizenship USA
(City , State) *(Country)*

P. O. Address 16817 South 1st Drive
Phoenix, Arizona 85045 USA

Full Name of Fourth/Joint Inventor (given name, family name) **Derrick C. Lin**

Inventor's Signature _____ Date _____

Residence Foster City, California USA Citizenship USA
(City , State) *(Country)*

P. O. Address 113 Barkentine Street
Foster City, California 94404 USA

Full Name of Fifth/Joint Inventor (given name, family name) **Francis X. McKeen**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 10612 N. W. LeMans Ct.
Portland, Oregon 97229 USA

Full Name of Sixth/Joint Inventor (given name, family name) **Gilbert Neiger**

Inventor's Signature _____ Date 7/27/2000

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 2424 N. E. 11th Avenue
Portland, Oregon 97212 USA

Full Name of Seventh/Joint Inventor (given name, family name) **Ken Reneris**

Inventor's Signature Ken Reneris Date 8/3/2006

Residence Wilbraham, Massachusetts USA Citizenship USA
(City , State) *(Country)*

P. O. Address 8 Red Gap Road
Wilbraham, Massachusetts 01095 USA

Full Name of Eighth/Joint Inventor (given name, family name) **James A. Sutton**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 20205 N. W. Paulina Drive
Portland, Oregon 97229 USA

Full Name of Ninth/Joint Inventor (given name, family name) **Shreekant S. Thakkar**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship United Kingdom
(City , State) *(Country)*

P. O. Address 150 S.W. Moonridge Place
Portland, Oregon 92775 USA

Full Name of Tenth/Joint Inventor (given name, family name) **Milland Mittal**

Inventor's Signature _____ Date _____

Residence Palo Alto, CA USA Citizenship USA
(City , State) *(Country)*

P. O. Address 800 E. Charleston Road, #29
Palo Alto, CA 94303 USA

Full Name of Seventh/Joint Inventor (given name, family name) **Ken Reneris**

Inventor's Signature _____ Date _____

Residence Wilbraham, Massachusetts USA (*City , State*) Citizenship USA (*Country*)

P. O. Address 8 Red Gap Road
Wilbraham, Massachusetts 01095 USA

Full Name of Eighth/Joint Inventor (given name, family name) **James A. Sutton**

Inventor's Signature _____ Date 7-24-2000

Residence Portland, Oregon USA (*City , State*) Citizenship USA (*Country*)

P. O. Address 20205 N. W. Paulina Drive
Portland, Oregon 97229 USA

Full Name of Ninth/Joint Inventor (given name, family name) **Shreekant S. Thakkar**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA (*City , State*) Citizenship United Kingdom (*Country*)

P. O. Address 150 S.W. Moonridge Place
Portland, Oregon 92775 USA

Full Name of Tenth/Joint Inventor (given name, family name) **Milland Mittal**

Inventor's Signature _____ Date _____

Residence Palo Alto, CA USA (*City , State*) Citizenship USA (*Country*)

P. O. Address 800 E. Charleston Road, #29
Palo Alto, CA 94303 USA

Full Name of Seventh/Joint Inventor (given name, family name) **Ken Reneris**

Inventor's Signature _____ Date _____

Residence Wilbraham, Massachusetts USA Citizenship USA
(City , State) *(Country)*

P. O. Address 8 Red Gap Road
Wilbraham, Massachusetts 01095 USA

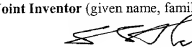
Full Name of Eighth/Joint Inventor (given name, family name) **James A. Sutton**

Inventor's Signature _____ Date _____

Residence Portland, Oregon USA Citizenship USA
(City , State) *(Country)*

P. O. Address 20205 N. W. Paulina Drive
Portland, Oregon 97229 USA

Full Name of Ninth/Joint Inventor (given name, family name) **Shreekant S. Thakkar**

Inventor's Signature  Date 8/7/00

Residence Portland, Oregon USA Citizenship United Kingdom
(City , State) *(Country)*

P. O. Address 150 S.W. Moonridge Place
Portland, Oregon 92775 USA

Full Name of Tenth/Joint Inventor (given name, family name) **Milland Mittal**

Inventor's Signature _____ Date _____

Residence Palo Alto, CA USA Citizenship USA
(City , State) *(Country)*

P. O. Address 800 E. Charleston Road, #29
Palo Alto, CA 94303 USA

Full Name of Tenth/Joint Inventor (given name, family name) **Milland Mittal**

Inventor's Signature [signature] Date 9/5/00

Residence Palo Alto, CA USA
(City, State)

Citizenship USA
(Country)

P. O. Address 800 E. Charleston Road, #29

Palo Alto, CA 94303 USA

Full Name of Eleventh/Joint Inventor (given name, family name) ______

Inventor's Signature ______ Date ______

Residence ______
(City, State)

Citizenship ______
(Country)

P. O. Address ______

______

Appendix A

I hereby appoint BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP, a firm including: William E. Alford, Reg. No. 37,764; Farzad E. Amini, Reg. No. 42,261; Aloysius T. C. AuYeung, Reg. No. 35,432, William Thomas Babbitt, Reg. No. 39,591; Carol F. Barry, Reg. No. 41,600; Jordan Michael Becker, Reg. No. 39,602; Lisa N. Benado, Reg. No. 39,995; Bradley J. Bereznak, Reg. No. 33,474; Michael A. Bernadicou, Reg. No. 35,934; Roger W. Blakely, Jr., Reg. No. 25,831; R. Alan Burnett, Reg. No. 46,149; Gregory D. Caldwell, Reg. No. 39,926; Andrew C. Chen, Reg. No. 43,544; Thomas M. Coester, Reg. No. 39,637; Donna Jo Coningsby, Reg. No. 41,684; Florin Corie, Reg. No. 46,244; Dennis M. deGuzman, Reg. No. 41,702; Stephen M. De Klerk, Reg. No. P46,503; Michael Anthony DeSanctis, Reg. No. 39,957; Daniel M. De Vos, Reg. No. 37,813; Robert Andrew Diehl, Reg. No. 40,992; Sanjeet Dutta, Reg. No. P46,145; Matthew C. Fagan, Reg. No. 37,542; Tarek N. Fahmi, Reg. No. 41,402; George Fountain, Reg. No. 37,374; Paramita Ghosh, Reg. No. 42,806; James Y. Go, Reg. No. 40,621; James A. Henry, Reg. No. 41,064; Willmore F. Holbrow III, Reg. No. P41,845; Sheryl Sue Holloway, Reg. No. 37,850; George W Hoover II, Reg. No. 32,992; Eric S. Hyman, Reg. No. 30,139; William W. Kidd, Reg. No. 31,772; Sang Hui Kim, Reg. No. 40,450; Walter T. Kim, Reg. No. 42,731; Eric T. King, Reg. No. 44,188; Erica W. Kuo, Reg. No. 42,775; George Brian Leavell, Reg. No. 45,436; Gordon R. Lindeen III, Reg. No. 33,192; Jan Carol Little, Reg. No. 41,181; Kurt P. Leyendecker, Reg. No. 42,799; Joseph Lutz, Reg. No. 43,765; Michael J. Mallie, Reg. No. 36,591; Andre L. Marais, under 37 C.F.R. § 10.9(b); Paul A. Mendonsa, Reg. No. 42,879; Clive D. Menezes, Reg. No. 45,493; Chun M. Ng, Reg. No. 36,878; Thien T. Nguyen, Reg. No. 43,835; Thinh V. Nguyen, Reg. No. 42,034; Dennis A. Nicholls, Reg. No. 42,036; Daniel E. Ovanezian, Reg. No. 41,236; Kenneth B. Paley, Reg. No. 38,989; Marina Portnova, Reg. No. P45,750; William F. Ryann, Reg. 44,313; James H. Salter, Reg. No. 35,668; William W. Schaal, Reg. No. 39,018; James C. Scheller, Reg. No. 31,195; Jeffrey Sam Smith, Reg. No. 39,377; Maria McCormack Sobrino, Reg. No. 31,639; Stanley W. Sokoloff, Reg. No. 25,128; Judith A. Szepesi, Reg. No. 39,393; Vincent P. Tassinari, Reg. No. 42,179; Edwin H. Taylor, Reg. No. 25,129; John F. Travis, Reg. No. 43,203; Joseph A. Twarowski, Reg. No. 42,191; Tom Van Zandt, Reg. No. 43,219; Lester J. Vincent, Reg. No. 31,460; Glenn E. Von Tersch, Reg. No. 41,364; John Patrick Ward, Reg. No. 40,216; Mark L. Watson, Reg. No. P46,322; Thomas C. Webster, Reg. No. P46,154; Steven D. Yates, Reg. No. 42,242; and Norman Zafman, Reg. No. 26,250; my patent attorneys, and Firasat Ali, Reg. No. 45,715; and Justin M. Dillon, Reg. No. 42,486; my patent agents, with offices located at 12400 Wilshire Boulevard, 7th Floor, Los Angeles, California 90025, telephone (714) 557-3800, with full power of substitution and revocation, to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith.